(12) UK Patent Application (19) GB (11) 2 280 765 (13) A

(43) Date of A Publication 08.02.1995

(21) Application No 9413165.3

(22) Date of Filing 30.06.1994

(30) Priority Data
(31) 05184918 (32) 27.07.1993 (33) JP

(71) Applicant(s)
Fujitsu Limited

(Incorporated in Japan)

1015 Kamikodanaka, Nakahara-ku, Kawasaki-shi, Kanagawa 211, Japan

(72) Inventor(s)
Yukihiro Nakagawa

(51) INT CL[6]
G06F 13/40 12/10

(52) UK CL (Edition N)
G4A AFD AFGDC AKA AKB1

(56) Documents Cited
EP 0194696 A2

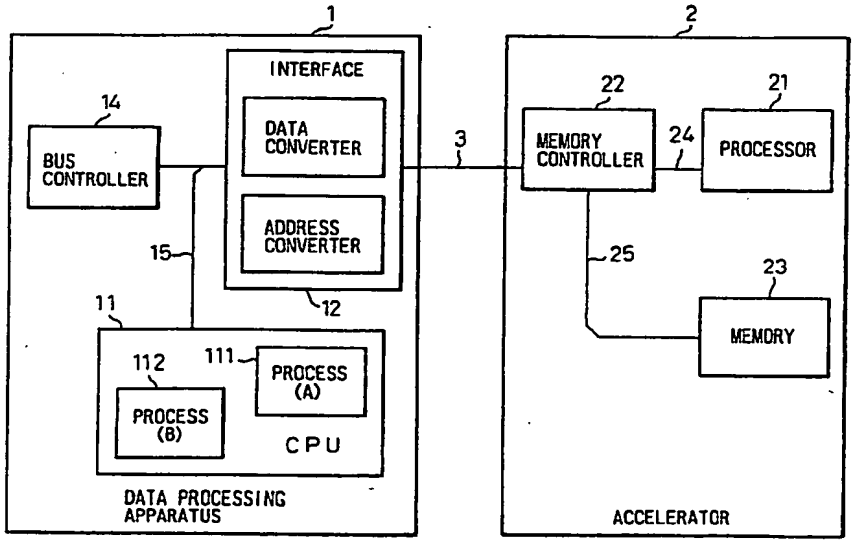
(58) Field of Search
UK CL (Edition M) G4A AFD AFGDC AKA AKB1 ANV
INT CL[5] G06F 12/10 13/40
ONLINE DATABASES : WPI

(74) Agent and/or Address for Service
Gill Jennings & Every
Broadgate House, 7 Eldon Street, LONDON, EC2M 7LH, United Kingdom

(54) **Multitasking data processing apparatus with different bus widths**

(57) A multitasking data processing apparatus (1) for connection to an external memory (23) having different data bus width, is arranged so that each process of a multitasking environment can freely access address space in the external memory for effecting packing. The data processing apparatus (1) is a multitasking data processing apparatus for concurrently executing a plurality of processes in a time-sharing method, and includes an internal bus (15) and an interface (12) to which an external memory (2) is connected in a form that the external memory (2) can be directly accessed from a CPU (11) of the apparatus via an external bus (3) having data width different from that of the internal bus (15). The interface (12) includes a data signal conversion circuit for converting data signals when the external memory (2) is accessed; and address signal conversion circuit for converting address signals when the external memory (2) is accessed. The address signal conversion circuit includes an address mapping table (AMT) indicating correlations between the internal bus addresses and the external bus addresses, and the correlations are optionally settable.

Fig.1



At least one drawing originally filed was informal and the print reproduced here is taken from a later filed formal copy.

This print takes account of replacement documents submitted after the date of filing to enable the application to comply with the formal requirements of the Patents Rules 1990.

GB 2 280 765 A

Fig.1

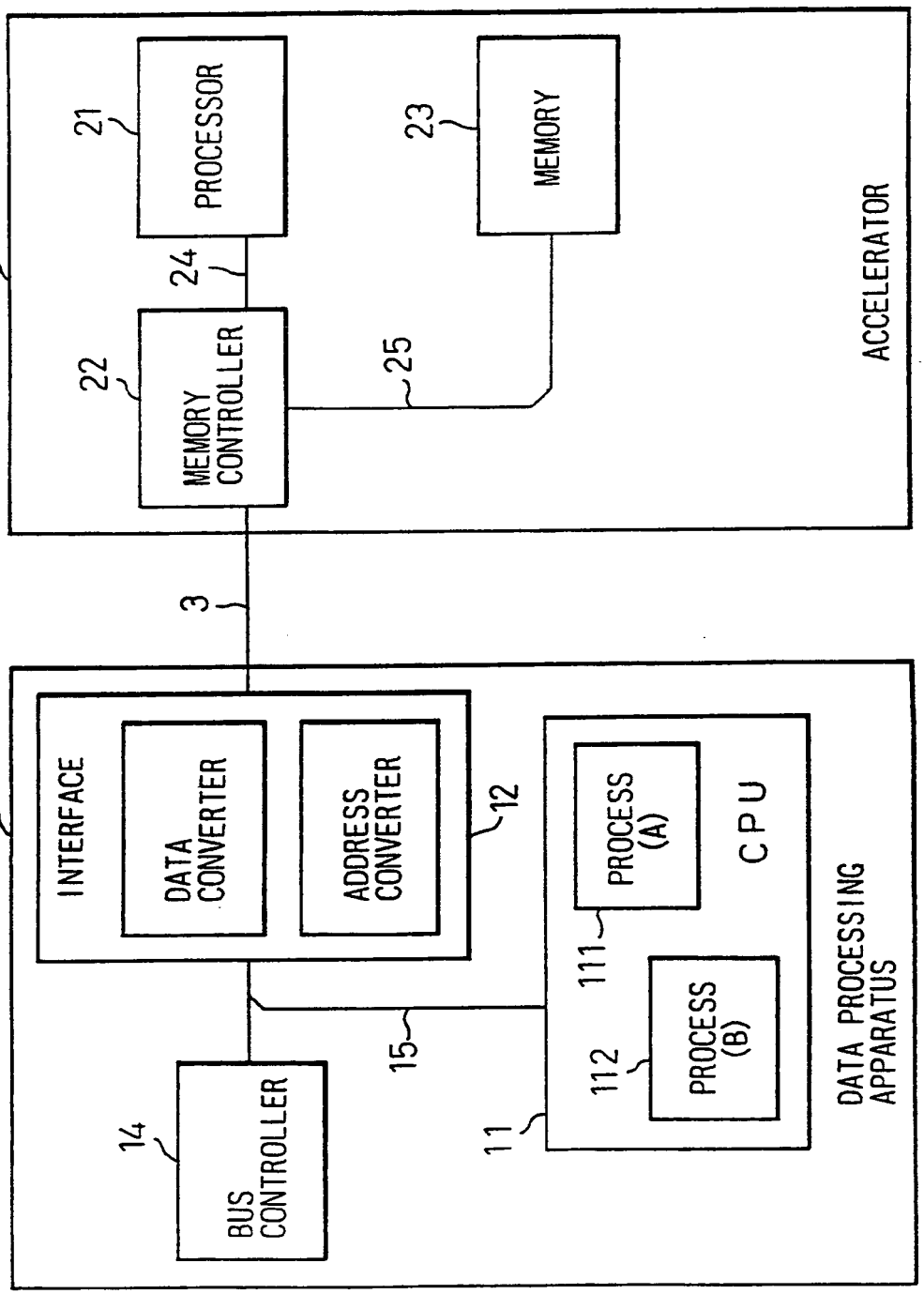


Fig. 2

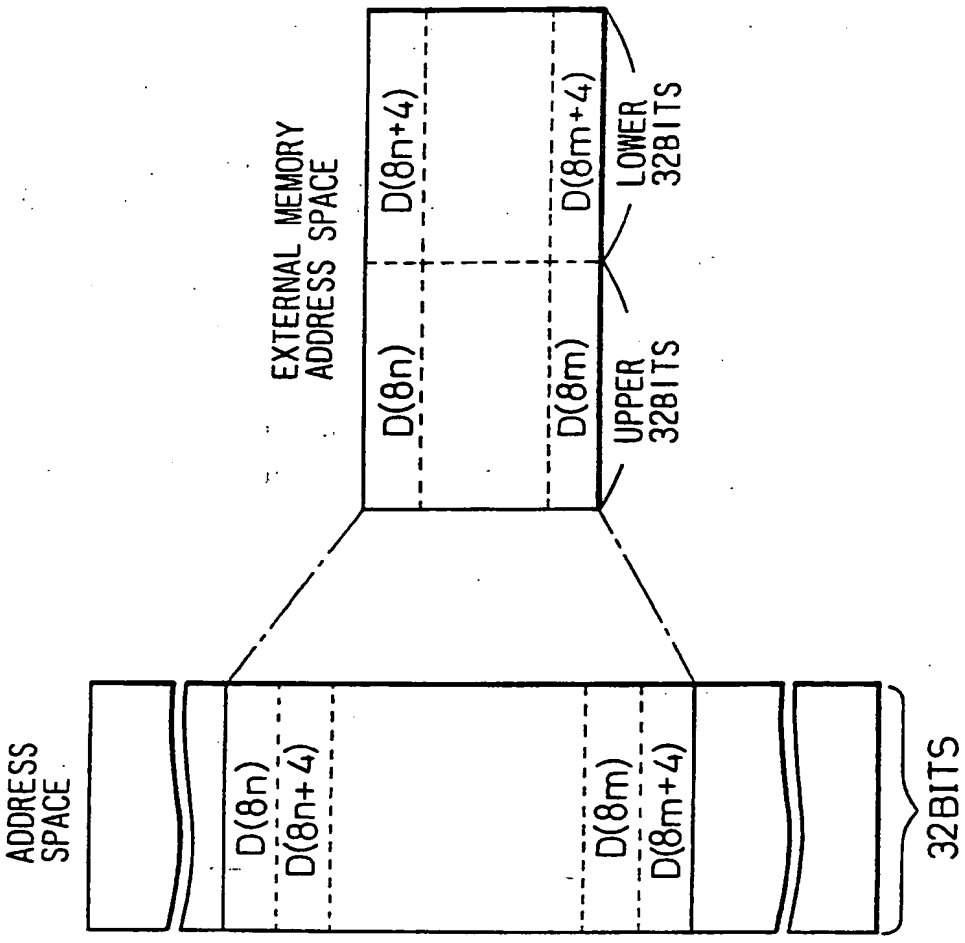


Fig.3
PRIOR ART

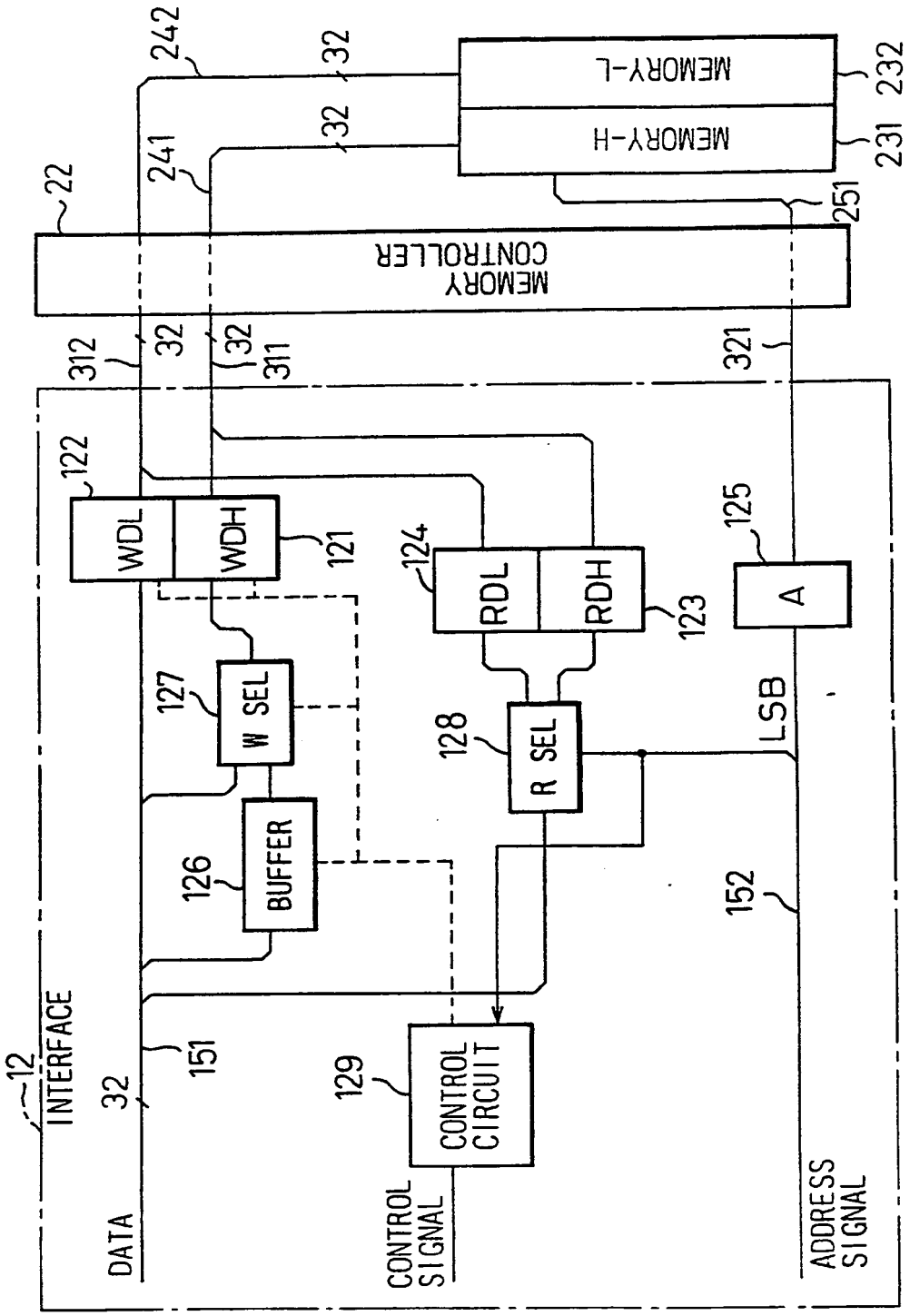


Fig.4
PRIOR ART

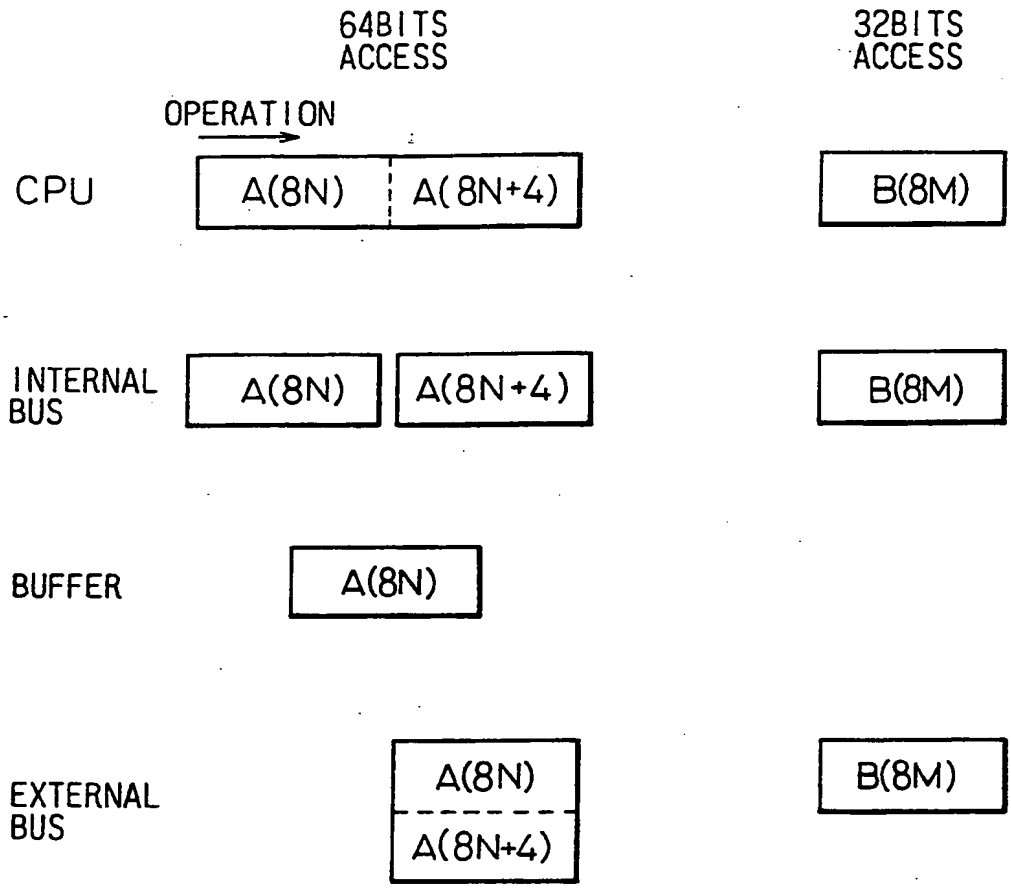


Fig.5
PRIOR ART

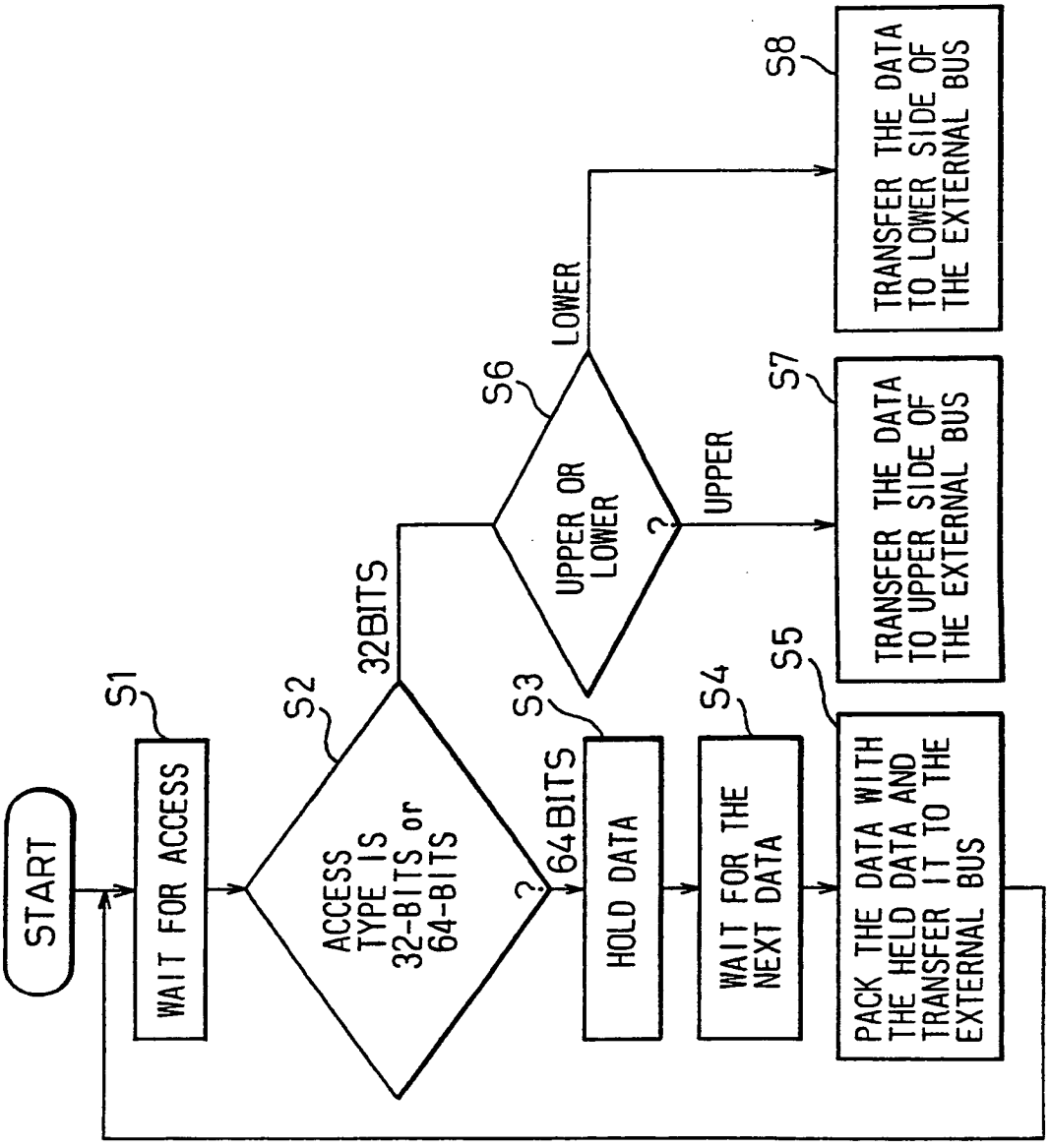


Fig.6
PRIOR ART

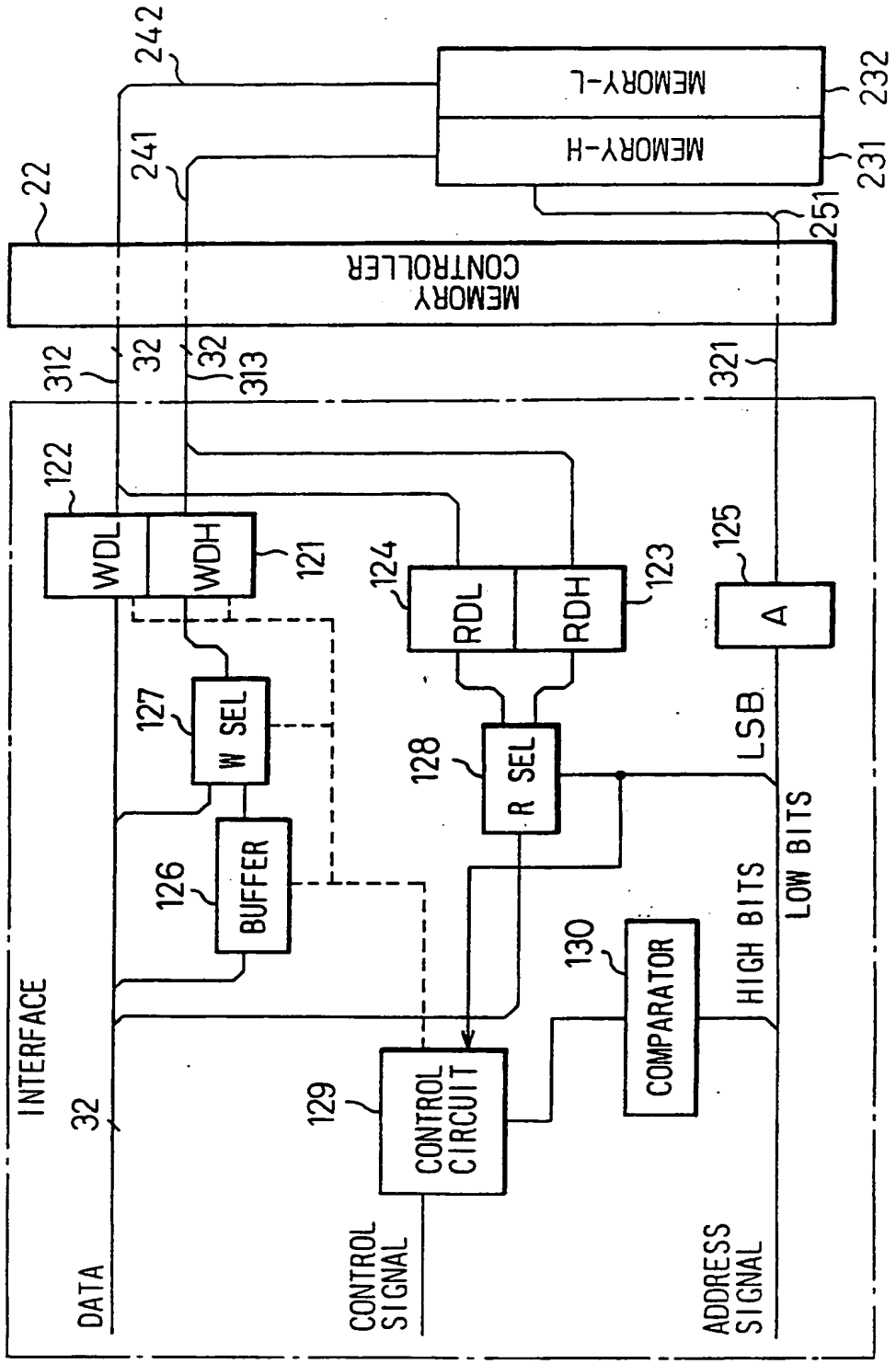


Fig.7A
PRIOR ART

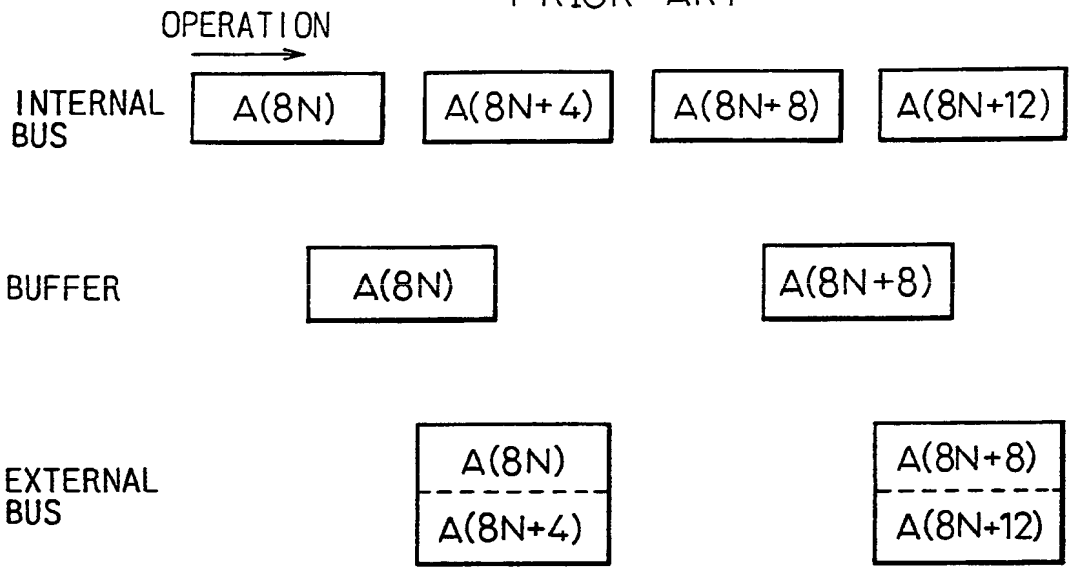


Fig.7B
PRIOR ART

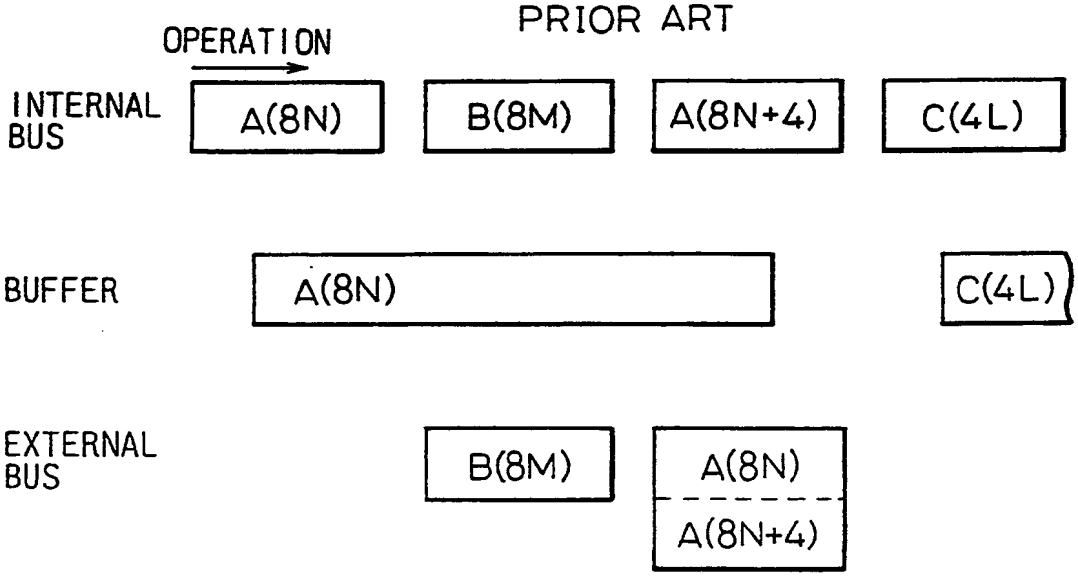


Fig.8
PRIOR ART

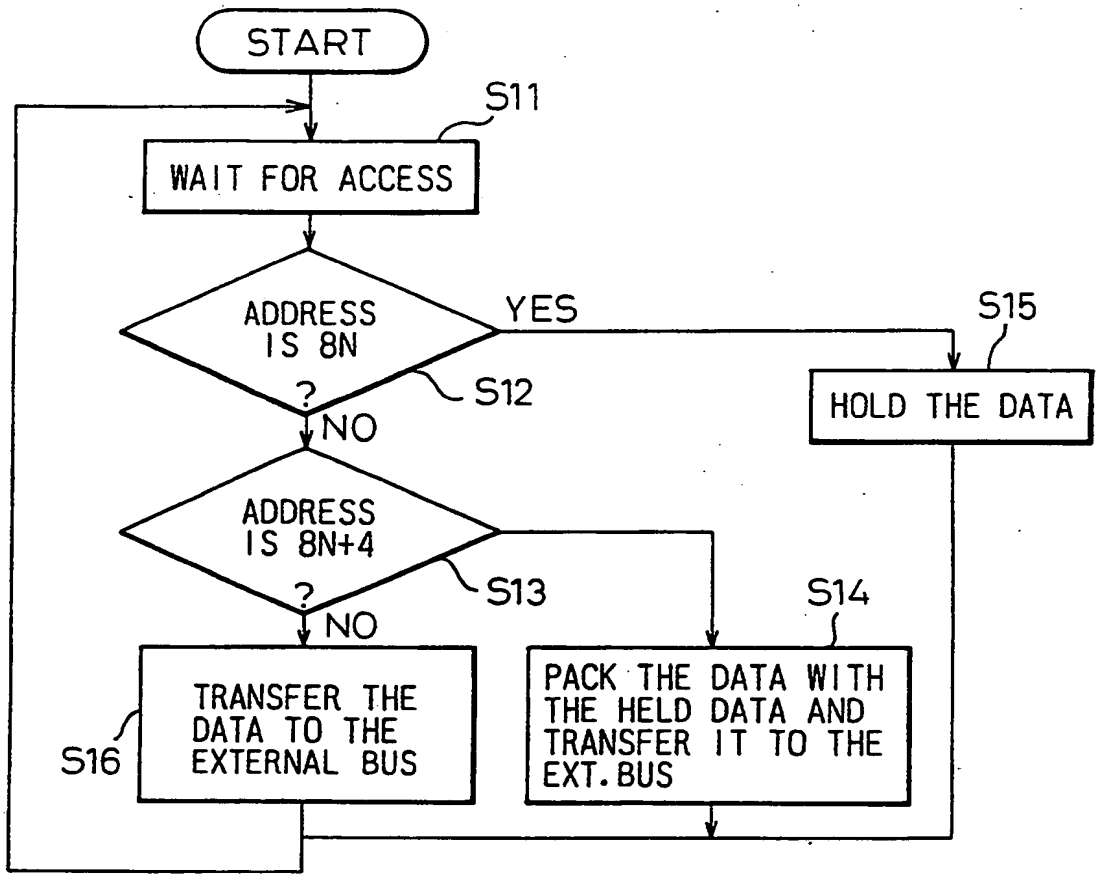


Fig.9

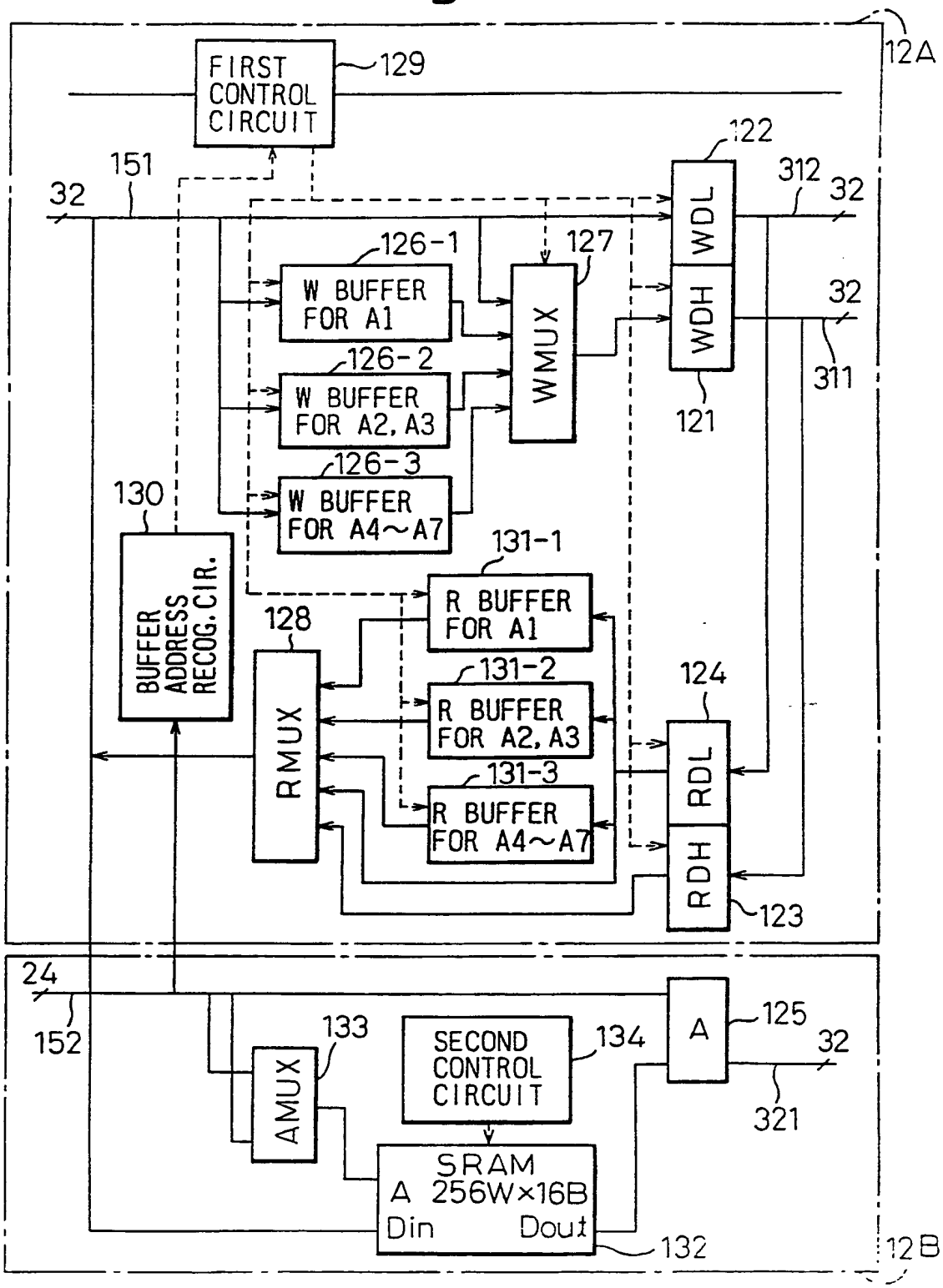


Fig.10

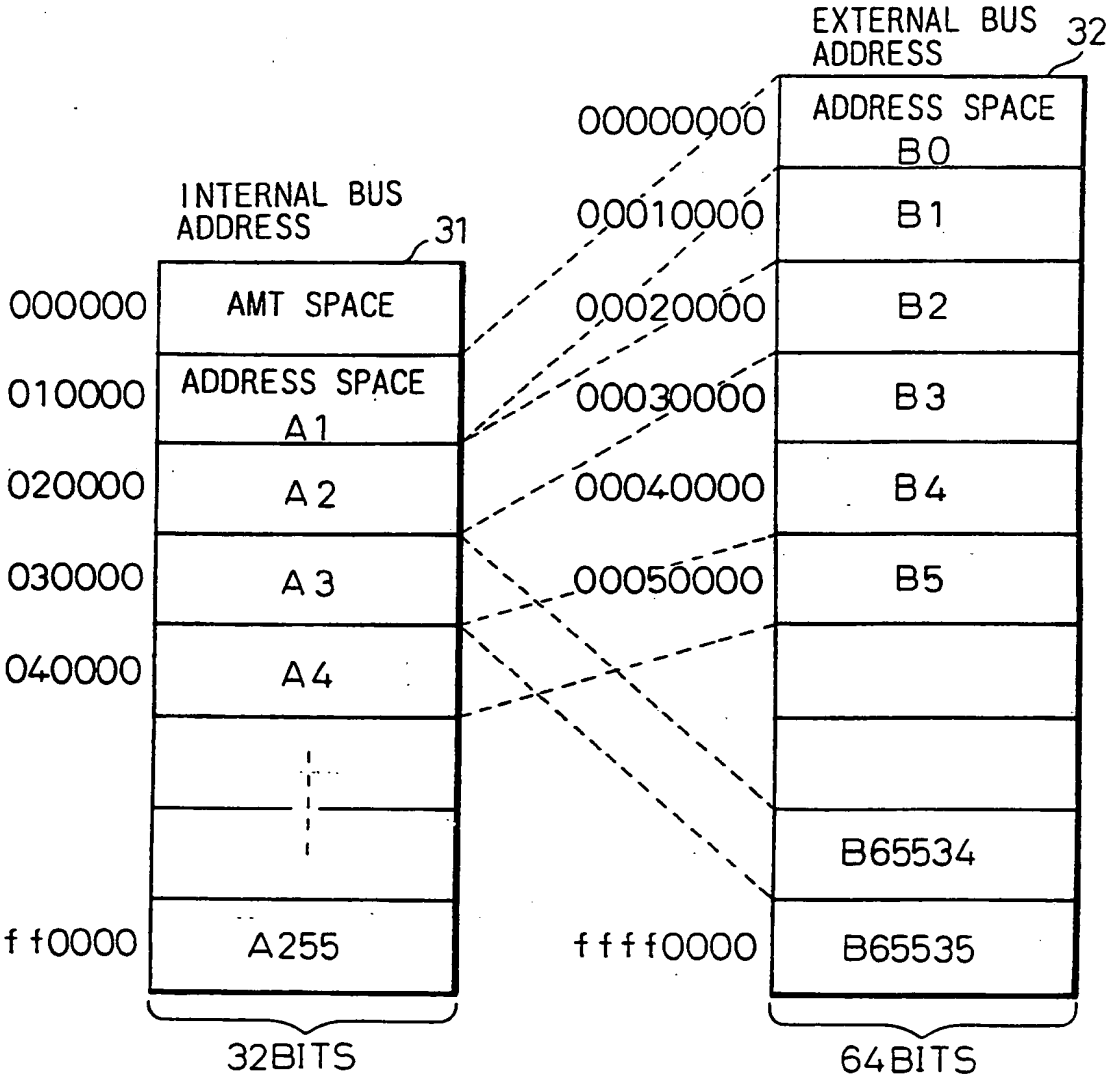


Fig.11

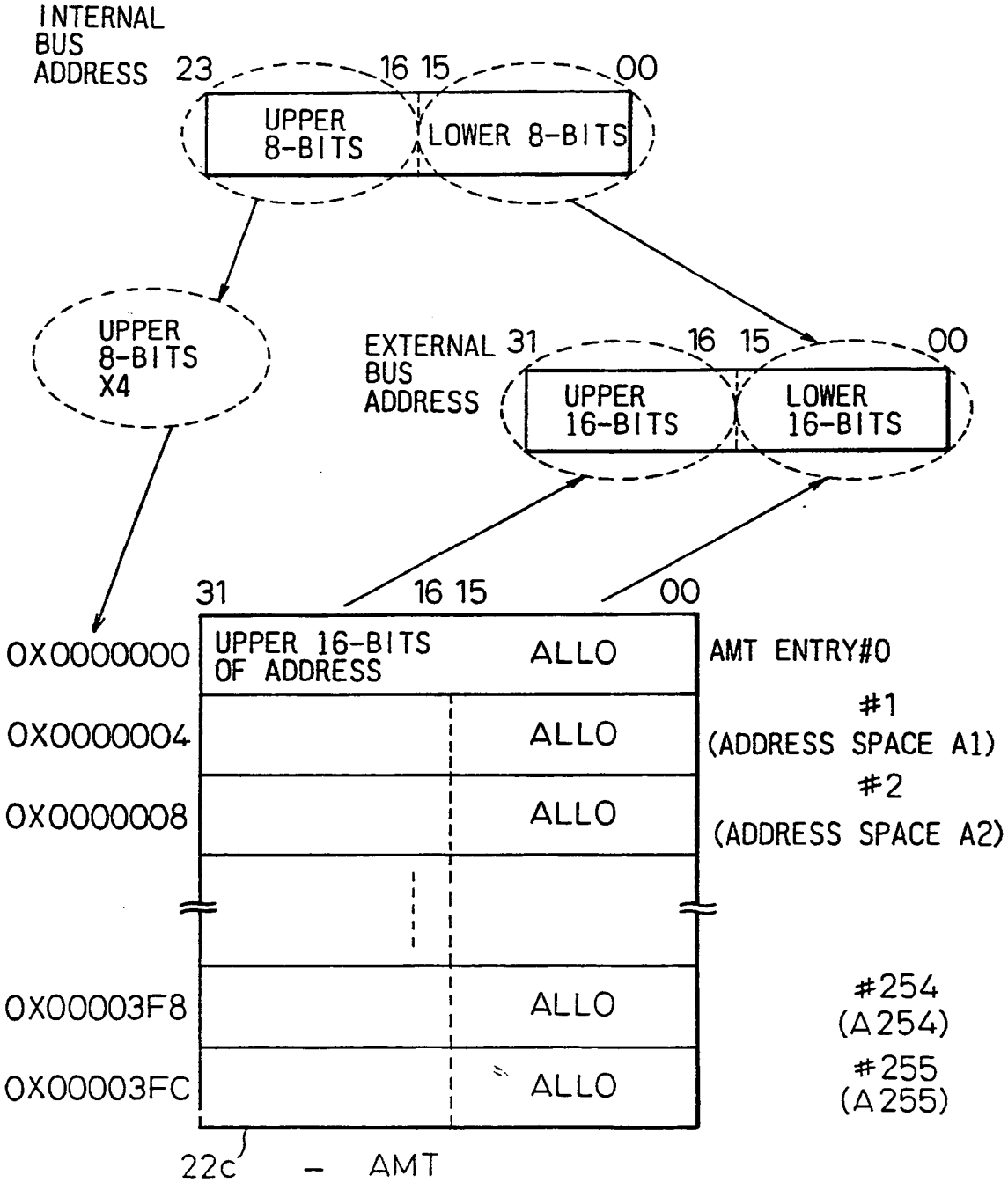


Fig.12

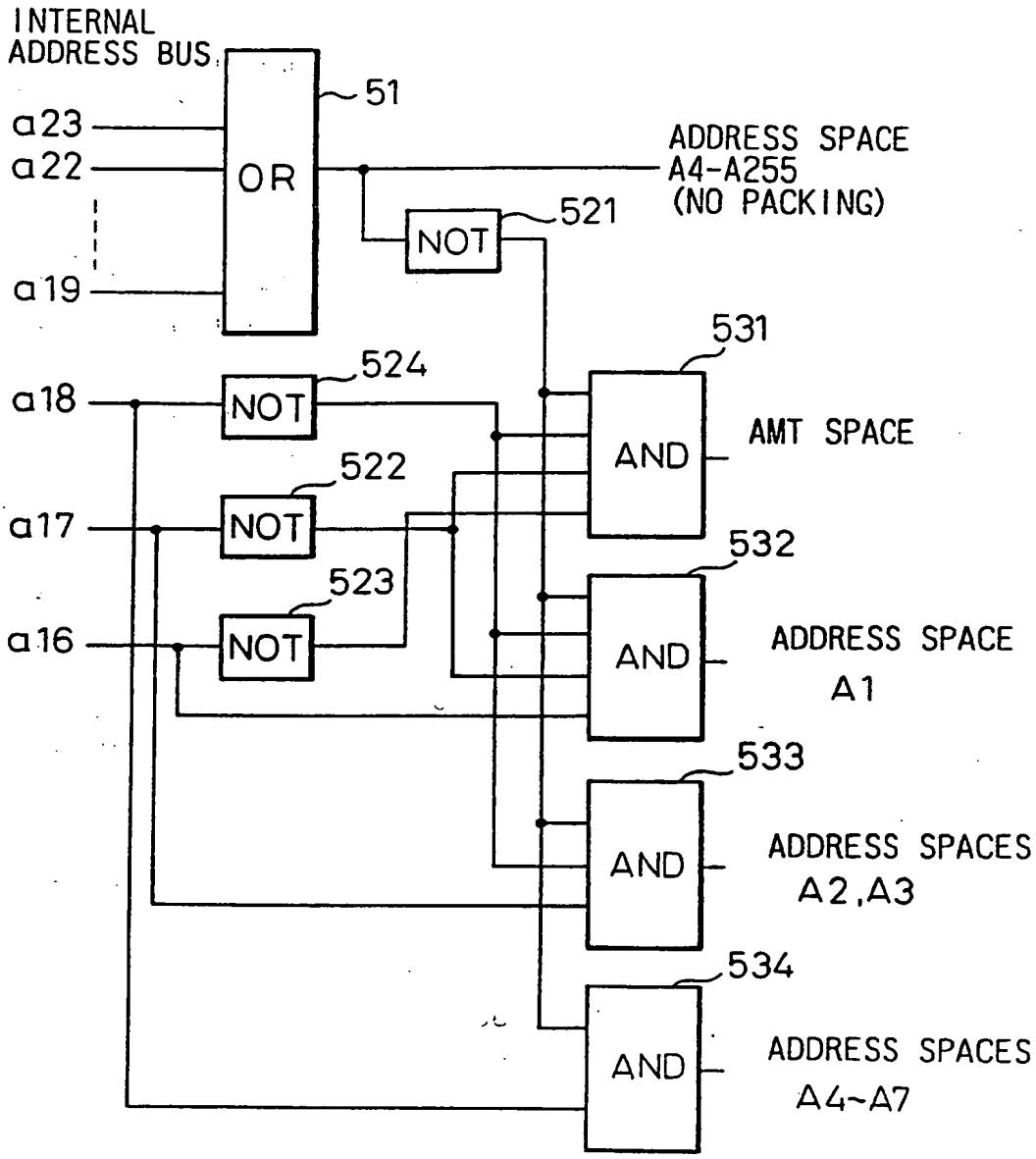


Fig.13

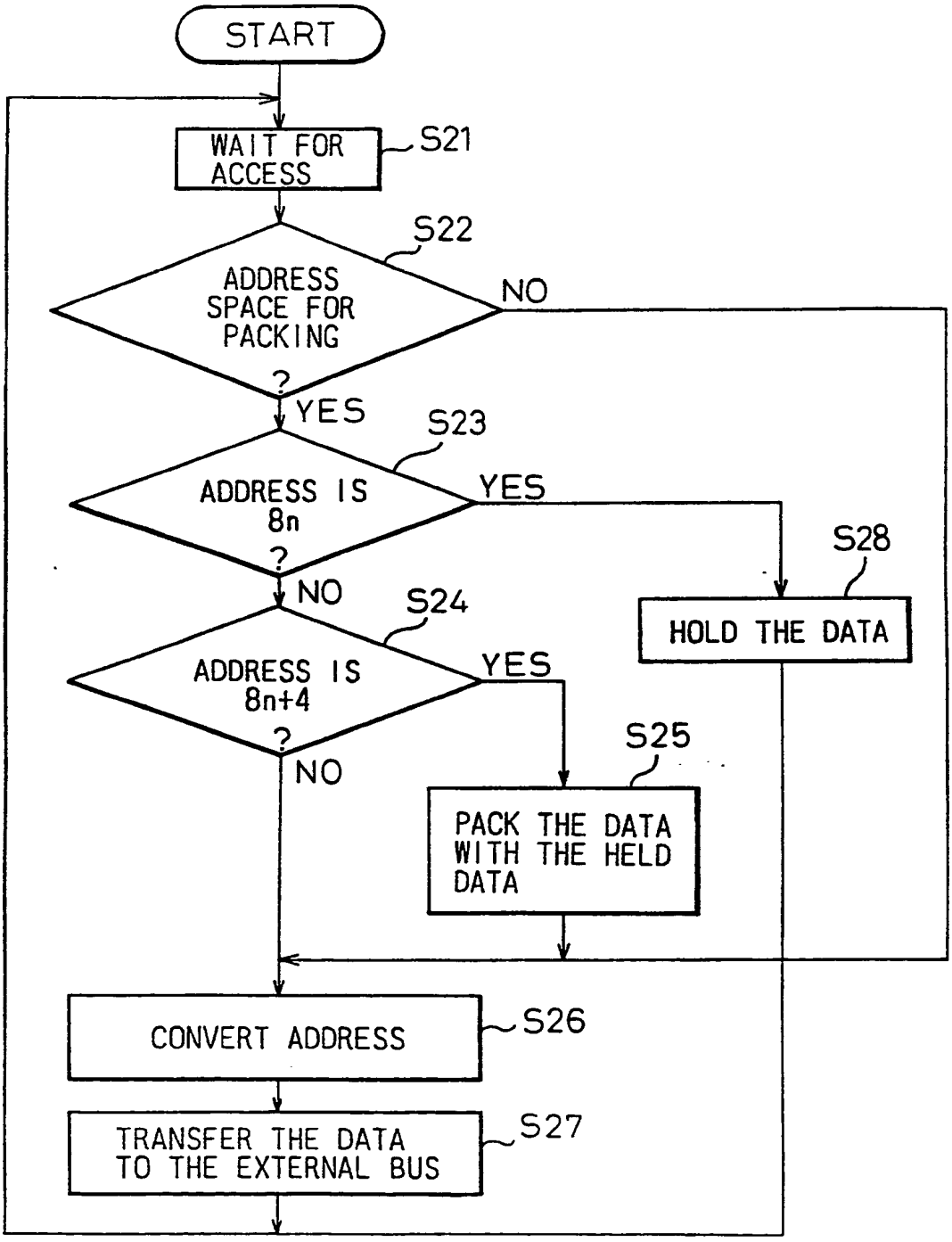
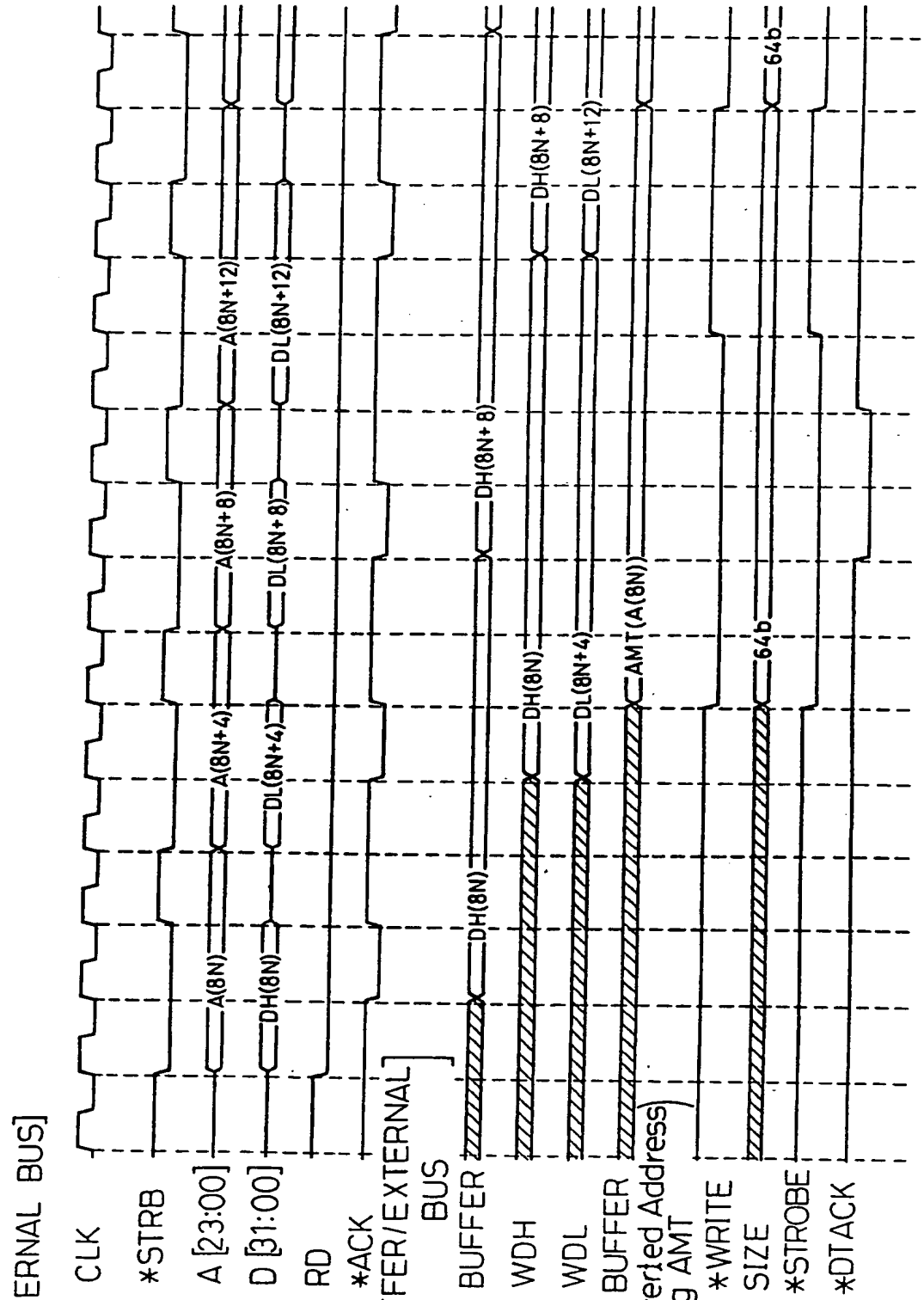


Fig.14



MULTITASKING DATA PROCESSING APPARATUS

The present invention relates to a data processing apparatus to which an external memory having data bus width different from that of an internal bus is connected in a form that the external memory can be directly accessed from a central processing unit (CPU) of the data processing apparatus. Particularly, the present invention relates to a multitasking data processing apparatus which concurrently executes a plurality of tasks (processes) in a time-sharing method. In the following, the data processing apparatus is called a computer.

In recent years, the possible size of an address space in a computer has become very large. In some computers, the capacities of memory devices provided within the computer realize only a part of total address space usable by the computer, and extra memory devices are externally connected via cables, etc., when a larger address space is necessary.

Further, shared memory is used in order to transfer data at high speed between computers. It is required that the shared memory can be directly accessed by both computers. Therefore, if the shared memory is provided in one computer, the shared-memory must be connected to the other computer via cables etc.

For example, the processing ability cf computers has increased in recent years. However, some work such as three-dimensional graphic display, etc., processes huge amounts of data, and normal host computers cannot fully process the data in an acceptable time. In the three-dimensional graphic display predetermined

graphical operations such as geometrical conversion calculations, color calculations, clipping operations etc., are repeated, and there exist specialized computers which perform these data processing operations at high speed. Generally, a specialized computer for graphic operations is called a CG (Computer Graphic) accelerator. The CG acelerator is used by connecting it to a host computer, and data transfer between the host computer and the CG accelerator is usually carried out via shared memory. In the following, an example in which a CG accelerator is connected to a host computer via shared memory will be described, however, the present invention is not limited to this example.

When the CG accelerator is connected to the host computer, the CG accelerator can be provided in the main body of the host computer or in a separate housing. When the the CG accelerator is provided in a separate housing, the CG accelerator is connected to the host computer via a cable.

The computer includes an internal bus. The internal bus is generally a synchronous bus over which data can be transferred at high speed. An interface is provided at the internal bus for transferring data to other outside devices. When the accelerator is provided in the main body of the computer, the accelerator can be directly connected to the internal bus, therefore, data transfer between the computer and the accelerator can be carried out at a speed corresponding to that of the internal bus. However, when the accelerator accomodated in the separate body is connected to the computer via the cable, the accelerator must be connected through the aforesaid interface. It is difficult to make the external bus in the cable a high speed synchronous type, therefore, the external bus is usually an asynchronous type or a low speed type. As a result, the transfer speed in the external bus is different from that of the

internal bus.

Consequently, when the accelerator accomodated in the separate body is connected to the host computer via a cable, the data transferred between the host computer and the accelerator must be converted so as to be adapted to the other type bus. Namely, when data is transferred from the host computer to the accelerator, internal bus data transferred at high speed must be converted so as to have a speed appropriate to be transferred in the external bus which is an asynchronous type or a low speed synchronous type. Therefore, a data transfer cycle in the external bus is longer that that of the internal bus.

Generally, an accelerator of a low performance range is small, therefore, it can be accomodated in the main body of the host computer. However, an accelerator of a high performance range is large, therefore, it must be accomodated in a separate housing. Consequently, it is connected via the cable. This means that data transfer between the host computer and the accelerator of the high performance range cannot be carried out at high speed although the accelerator of the high performance range usually requires huge amount of data to be transferred. As a result, there occurs a problem that the ability of the accelerator cannot be fully used since the data transfer speed is low. It has been desired to develop technology for transferring data at high speed between the data processing devices having separate housings.

In order to increase the data transfer speed, it can be contrived to increase the data bus width of the external bus thereby to increase the amount of data that can be transferred at one time. Here, however, a problem arises in regard to how the discrepancy be absorbed between the data bus width of the internal system bus and the data bus width of the external bus in the multitasking environment.

For example, when the data bus width of the internal bus is 32bits and the data bus width of the external bus is 64bits, it is now presumed that two sets of data each having 32 bits are combined into one set of data (this operation is called packing) and are transferred. In a singletasking environment in which operations of only one process are sequentially executed, a method can be employed in which when first word size data is written onto an address (8n), the first data is held in a buffer and when second word size data is written onto an address (8n + 4), the first data and second data are transferred together onto the external bus. Namely, 64-bit data consisting of the first and second data are transferred on the external bus. In a multitasking environment, on the other hand, another process may use the internal bus after the first word size data is transferred but before the next word size data is transferred. There occurs a problem in that the data are not properly transferred. Therefore, the same method as that of the case of the singletasking environment cannot be adapted in the multitasking environment.

The following two methods have hitherto been known to transfer data between the internal and external buses having different data bus width in the multitasking environment.

In the first conventional method, access from the CPU to the external memory is made twice consecutively from the same process. The firstly transferred data is held in the buffer and is output to the external bus together with the secondarily transferred data. This method is usable when the CPU consecutively outputs each set of data at any time. For example, this situation is realized when the CPU uses 64-bit data inside of the CPU and outputs each set of 32-bit data, obtained by dividing the 64-bit data, to the internal bus. More strictly speaking, the first conventional

method can be adapted only when a plurality of access operations of one process to the external memory are consecutively carried out. Therefore, number of computer systems to which this method can be adapted is limited.

In the second method, sets of addresses, for effecting the packing, are previously determined in an address space and, when the first one of each set of the above addresses is accessed from the CPU, the data is held, and when the second one of the set is accessed, the two words of data are packed into one to effect a double word transfer. Here, however, the above addresses are previously determined to be accessd from only one process. In this method, the addresses for effecting the packing are previously determined in an address space, therefore, there arises a problem that the number of addresses that each process can access are limited.

An object of the present invention is to improve the above-mentioned second method and to realize a data processing apparatus, connected with an external memory having different data bus width, in which apparatus each process of a multitasking environment can freely access address space from the external memory to effect packing.

The data processing apparatus according to a first aspect of the present invention is a multitasking data processing apparatus for concurrently executing a plurality of processes in a time-sharing mode, and comprises an internal bus; and an interface to which an external memory is connected in a form that the external memory can be directly accessed from a central processing unit of the apparatus via an external bus having data bus width different from that of the internal bus. The interface includes data signal conversion means for converting data signals when the external memory is accessed and address signal

conversion means for converting address signal when the external memory is accessed. The address signal conversion means includes an address mapping table indicating correlations between the internal bus addresses and the external bus addresses, and the correlations are optionally settable.

In the data processing apparatus according to the first aspect of the present invention, since the correlations between the internal bus addresses and the external bus addresses are optionally settable by the address mapping table, address space corresponding to the external memory for effecting the data packing can be freely accessed from each process of the multitasking environment.

The data processing apparatus according to a second aspect of the present invention, in addition to the constitution of the first aspect except the address mapping table, further includes a plurality of data packing means for packing internal bus data, and the address space of the external memory is divided into a plurality of blocks, and the plurality of data packing means are respectively allocated to the blocks.

In the data processing apparatus according to the second aspect of the present invention, when processes of the multitasking environment are respectively allocated to the blocks of the address space, transfer data for each process can be independently packed by using the data packing means allocated to its block, therefore, the processes to be executed can be optionally changed.

An example of the present invention will now be described with reference to the accompanying drawings, wherein:-

Fig. 1 is a diagram showing an example of the constitution of a data processing apparatus to which the present invention is applied;

Fig. 2 is a diagram showing address space of a conventional apparatus to which an external memory having double data bus width is connected;

Fig. 3 is a diagram showing a constitution of an apparatus which realizes a multitasking environment according to a first conventional method when an external memory having double data bus width is connected;

Fig. 4 is a diagram showing data output conditions on an internal bus and in an external bus during memory accessing operations when the first conventional method is carried out in the apparatus shown in Fig. 3;

Fig. 5 is a flowchart showing operations when the first conventional method is carried out in the apparatus shown in Fig. 3;

Fig. 6 is a diagram showing a constitution of an apparatus which realizes a multitasking environment according to a second conventional method when an external memory having double data bus width is connected;

Figs. 7A and 7B are diagrams showing data output conditions on an internal bus and an external bus in the memory accessing operations when the second conventional method is carried out in the apparatus shown in Fig. 6;

Fig. 8 is a flowchart showing operations when the second conventional method is carried out in the apparatus shown in Fig. 6;

Fig. 9 is a diagram showing a constitution of an interface of an embodiment of the present invention;

Fig. 10 is a diagram showing address space of the embodiment;

Fig. 11 is a diagram explaining address converting operation in the embodiment;

Fig. 12 is a diagram showing a constitution of an address discrimination circuit of the interface shown in Fig. 9;

Fig. 13 is a flowchart showing operations in the embodiment;

Fig. 14 is a timechart showing an example of signal conditions of elements in the embodiment.

Before proceeding to a detailed description of the preferred embodiments of the present invention, a prior art data processing apparatus will be described to allow a clearer understanding of the differences between the present invention and the prior art.

Fig. 1 is a diagram showing an example of data processing apparatuses (computers) to which the present invention is applied. In Fig. 1, a CG accelerator is connected to a computer which operates as a host computer. In the following, examples in which the present invention is applied to a host computer connected to a CG accelerator will be described. However, the present invention is not limited to these examples. The present invention can be applied to any apparatus in which memories having different data bus widths are connected to the external bus in forms that those memories are directly accessed. In figures shown in the following, the same reference numerals will be attached to portions having same functions.

In Fig. 1, reference numeral 1 indicates a main portion of the computer which is a Work Station (WS) in this embodiment; 2 indicates a CG accelerator which is connected to the WS 1; 11 indicates a CPU of the WS1; 12 indicates an interface to which the CG accelerator 2 is connected; 14 indicates a bus controller; 15 indicates an internal bus whose data bus width is 32-bits; 3 indicates an external bus whose data bus width is 64-bits; 21 indicates a processor in the CG accelerator; 22 indicates a memory controller; 23 indicates shared memory in the CG accelerator 2; 24 and 25 indicate accelerator buses whose data bus width is 64-bits; and 111 and 112 indicate processes which are executed at the

same time.

A huge number of operations are required for processsing three dimensional graphic data. The CG accelerator 2 receives a huge amount of data, consisting of a three dimensional graphic model, from the host computer 1, and operations are carried out on the received data. For example, a two dimensional image viewed from a visual point is generated by the CG accelerator 2. In this way, the number of operations carried out in the WS 1 is reduced. There are several methods to transfer data between the WS 1 and the CG accelerator 2. In the constitution shown in Fig. 1, the shared-memory 23 provided in the CG accelerator 2 is used for this purpose, and data are transferred between the host computer 1 and the CG accelerator 2 by accessing the memory 23 from the WS 1. The memory controller 22 controls the access to the memory 23 whether it is accessed by the WS 1 or by the processor 21. When the memory 23 is accessed from the WS 1, the CPU 11 can directly access the memory 23 via the internal bus 15, the interface 12 and the external bus 3.

As described above, although the internal bus 15 is a synchronous type bus in which data can be transferred at high speed, the external bus 3 is an asynchronous type bus or a low-speed synchronous type bus. Therefore, data cannot be transferred at high speed in the external bus 3, and the transfer cycle of the external bus is longer than that of the internal bus. Therefore, it can be contrived to increase the data bus width of the external bus 3 thereby to increase the amount of data that can be transferred at one time. In the system shown in Fig. 1, the external 64-bit data bus is connected to the internal 32-bit data bus 15. When WS 1 writes data into the memory 23, two sets of 32-bit data are combined into one set of 64-bit data, and the 64-bit data are written into the memory 23 via the external bus 3. This operation is called a packing operation.

When WS 1 reads data from the memory 23, 64-bit data which is read from the memory 23 via the external bus 3 are held in a buffer, then the CPU 11 reads the data, 32-bits at a time in two read operations. As described later, in normal systems, the memory 23 is also accessible in the form of 32-bit data, therefore, the memory 23 is directly accessed from the CPU 11 in the 32-bit data form when the low access speed causes no problem. The problem generated in the read operation is same to those in the write operation, and the access speed becomes a problem particularly when WS 1 writes data into the memory 23. Therefore, in the following, examples in the write operations will be explained.

Fig. 2 is a diagram showing address space in the apparatus shown in Fig. 1. The left portion shows address space in the internal bus of 32-bit data bus width, and areas indicated by references D(8n) to D(8m+4) are address space corresponding to the memory 23. The right portion indicates address space in the memory 23, namely, address space in the external bus. As shown in Fig. 2, each set of 32-bit data on the contiguous addresses in the address space of the 32-bit bus are combined into 64-bit data.

For example, when two words of data are packed, first data D(8n) which are written into an address (8n) are held in the buffer. When second data D(8n+4) which are written into an address (8n+4) are output to the internal bus, the first data D(8n) and the second data D(8n+4) are output together to the external bus. When this method is carried out in a singletasking environment in which each process is sequentially carried out, there no problem arises. However, when this method is carried out in a multitasking environment in which a plurality of processes are concurrently carried out in a time-sharing method, there arises a problem that two data cannot be correctly packed. For example, this situation can arise when a plurality of windows in

which three dimensional graphic images are respectively displayed. It is assumed that the currently executing process is changed from a first process to a second process after the first process outputs first data for packing (first packing data), then the first process returns to be executed again after the second process outputs several data for packing. When the first process is executed again, the first process will output second packing data, however, the first packing data stored in the buffer is already destroyed. Namely, the data held in the buffer is not the data to be packed with the second packing data. Therefore, if the data held in the buffer is would be packed with the second data, data cannot be correctly transferred. This means that new packing methods different from those in the singletasking environment are necessary in the multitasking environment.

The following two methods have hitherto been known to transfer the packed data in the multitasking environment.

In the first method, access from the CPU to the external memory is made twice consecutively by the same process. The data transferred during the first access are held in the buffer and are output to the external bus together with the data transferred during the second access.

Fig. 3 is a diagram showing a constitution of an interface by which the above first conventional method is carried out.

In Fig. 3, reference numeral 12 indicates an interface; 22 indicates a memory controller; 231 indicates a memory which stores the upper 32-bit data of 64-bit data; 232 indicates a memory which stores lower 32-bit data of the 64-bit data; 311 and 312 indicate 32-bit data buses consisting of a 64-bit external data bus; 321 indicates an external address bus; 241 and 242 indicate 32-bit data buses connected

between the memory controller 22 and the memories 231 and 232; 251 indicates an address bus. The memories 231 and 232 can be independently set in enable states. Namely, data can be simultaneously read from or written into both memories or only either one of the memories. When the memories 231 and 232 are accessible from the WS, the data buses 241 and 242 are substantially connected direct to the data buses 311 and 312, and the address bus 251 is also connected direct to the address bus 321.

Reference numeral 121 indicates a first write buffer which holds 32-bit data output to the external data bus 311; 122 indicates a second write buffer which holds 32-bit data output to the external data bus 312; 123 indicates a first read buffer which is used for reading 32-bit data on the external data bus 311; 124 indicates a second read buffer which is used for reading 32-bit data on the external data bus 312; 125 indicates an address buffer which outputs address signal to the external address bus 321 when the WS accesses the external memories 231 and 232; 126 indicates a packing buffer which holds the first 32-bit data to be packed; 127 indicates a write multiplexer which selects either of data on the internal data bus 151 or data held in the packing buffer 126, and outputs the selected data to the first write buffer 121; 128 indicates a read multiplexer which selects either of data output from the first read buffer 123 or the second read buffer 124 and outputs the selected data; and 129 indicates a control circuit of the interface.

Two 32-bit data held in the first write buffer 121 and the second write buffer 122 consist of the packed 64-bit data. The first and second write buffers 121 and 122 can latch the data. The first and second read buffers 123 and 124 can also latch data. Address signal of the internal address bus except a set of upper bits which indicate a block of address space corresponding to the external memories 231 and 232 are output to the

external address bus 321.

Fig. 4 is a diagram showing data output conditions on the internal bus and the external bus in the memory accessing operations when the above-mentioned first method is carried out in the apparatus shown in Fig. 3. Fig. 5 is a flowchart showing operations when the first conventional method is carried out in the apparatus shown in Fig. 3. The operations in the first conventional method will be described with reference to Figs. 4 and 5.

As shown in Fig. 4, when 64-bit data is written into the external memories from the CPU, the CPU consecutively outputs two data A(8n) and A(8n+4) to the internal data bus. The data A(8n) and A(8n+4) are respectively written into the addresses (8n) and (8n+4). The packing buffer 126 holds the first data A(8n) which is first output, and the write multiplexer 127 outputs the first data A(8n) held in the packing buffer 126 to the first write buffer 121. Next, the second data A(8n+4) are output to the internal data bus, then the second data A(8n+4) are output to the external data bus 312 from the second write buffer 122. At this time, the first data A(8n) are output to the external data bus 311 from the first write buffer 121. Therefore, two data A(8n) and A(8n+4) are simultaneously output to the external bus, and these two data are written into the memories 231 and 232.

Further, only one 32-bit data can be written into either of the memories 231 and 232 being set in enable state. When the 32-bit data A (8n) is written into the address (8n), the first write buffer 121 is selected, and the 32-bit data A (8n) is written into the memory 231. When the data A (8n+4) is written into the address (8n+4), the second write buffer 122 is selected, and the data A (8n+4) is written into the memory 232.

As shown in Fig. 5, at step S1, the interface waits for access to the external memory. When the CPU outputs

data to be written into the external memory, the access type is judged at step S2. When the access type is judged to be 64-bits, operations from step S3 to S5 are carried out. At step S3, the packing buffer 126 holds the data on the internal bus, and the multiplexer 127 is set to output the data in the packing buffer 126. At step S4, the interface waits for the next data. At step S5, two data are output from the first and second write buffers 121, 122 to the external data buses 311, 322. When the access type is 32-bits, an operation of step S6 is carried out. At step S6, it is determined that the data is written into the memories 231 or 232 according to the least significant bit of the address signal. When the data are written into the memory 231, step S7 is carried out. When the data are written into the memory 232, an operation of step S8 is carried out. At step S7, the write multiplexer 127 is set to output the data on the internal bus 151, and the data are output to the external bus 311. The external memory 231 is set to be enable, therefore, the data are written only into the memory 231. At this time, the external memory 232 is disabled, therefore, the data in the memory 232 do not change. Similarly, at step S8, the second write buffer 122 outputs the data of the internal bus 151 to the external bus 312. The external memory 232 is set to be enable, therefore, the data are written only into the memory 232. At this time, the external memory 231 is disabled, therefore, the data in the memory 231 do not change.

As described above, the first conventional method can be carried out when access from the CPU to the external memory is made twice consecutively from the same process. For example, there is a computer system in which a CPU handles 64-bit data in the CPU, and the CPU divides the 64-bit data into two 32-bit data for outputting them to the internal bus. In this computer system, the external memory is accessed by a unit of

64-bits although the practical access is carried out as a set of two consecutive 32-bit data. Therefore, in this computer system, the process is not switched to another process when the first 32-bit data is output to the internal bus from the CPU. As a result, there arises no problem when the first conventional method is carried out in this computer system.

However, in normal computer system in which the CPU handles 32-bit data in the CPU and 32-bit data are output to the internal bus, it is difficult to control the switching operations of processes. If the switching operations of the processes is to be controlled, the operating system of the computer system becomes complicated. Therefore, in computer systems which are controlled by normal operating systems, it is difficult to arrange that the operations of each process consecutively access the external memory at any time.

In the second conventional method, addresses for effecting the packing are set in an address space corresponding to the external memory. When the above addresses are accessed from the CPU, it is judged whether or not the accessed address is included in the addresses for the packing. When the addresses for the packing are accessed, the two word data are packed into one to effect double word transfer. Here, however, the addresses for effecting the packing are accessed by only one process.

Fig. 6 is a diagram showing a constitution of an interface by which the above second conventional method is carried out.

By comparing Fig. 6 with Fig. 3, it is apparent that the constitution of Fig. 6 is almost same as that of Fig. 3 except that a comparator 130 is further provided. The comparator 130 judges whether an address signal on the internal address bus is (8N) or (8N+4). When the comparator 130 detects that the address (8N) is accessed, the comparator 130 outputs a detection

signal to the control circuit 129.

Figs. 7A and 7B are diagrams showing data output conditions on the internal bus and the external bus in the memory access operations when the above-mentioned second method is carried out in the apparatus shown in Fig. 6. Fig. 7A shows data output conditions when addresses for effecting the packing are consecutively accessed, and Fig. 7B shows data output conditions when an address not effecting the packing is accessed between first and second access operations effecting the packing. Fig. 8 is a flowchart showing operations when the second conventional method is carried out in the apparatus shown in Fig. 6. The operations in the second conventional method will be described with reference to Figs. 7A, 7B, and Fig. 8.

As shown in Fig. 7A, when addresses for effecting the packing are consecutively accessed, data A(8N) and A(8N+4) are consecutively output to the internal data bus. The data A(8N) indicate data written into an address (8N), and the data A(8N+4) indicate data written into an address (8N+4). When the comparator 130 detects that the address (8N) is accessed, the comparator 130 outputs its judgement result to the control circuit 129. In response to this judgement result, the control circuit 129 controls the packing buffer 126 to latch the data on the internal data bus. Next, the comparator 130 detects that the address (8N+4) is accessed, the comparator 130 outputs its judgement result to the control circuit 129. In response to this judgement result, the control circuit 129 controls the write multiplexer 127 to output data held in the packing buffer 126, and further controls the first and second write buffers 121 and 122 to output two 32-bit data to the external data buses 311 and 312. In this way, a set of two 32-bit data are written into the external memories 231 and 232.

As shown in Fig. 7B, when a different access

operation to access an address not effecting the packing is carried out after the first packing address is accessed, operations same to those of Fig. 7A are carried out except that the data A(8N) is held in the packing buffer 126 during this different operation.

In the second conventional method, when an address not the packing is accessed, operations the same as those of the 32-bit data access operations are carried out.

In the second conventional method, as shown in Fig. 8, at step S11, the interface waits for access to the external memory. When the CPU outputs data to be written into the external memory, at step S12, it is judged whether the address of this access operation is included in the addresses for effecting the packing and is (8N). When the address is included in the addresses for effecting the packing and is (8N), the control proceeds to step S15, and the data on the internal bus are latched in the packing buffer 126. When the address is not included in the addresses for effecting the packing and is not (8N), at step S12, it is further judged whether the address of this access operation is included in the addresses for effecting the packing and is (8N+4). When the address is included in the addresses for effecting the packing and is (8N+4), the control proceeds to step S14, and the data are output to the external bus together with the data held in the packing buffer 126. When the address of this access operation is not included in the addresses for effecting the packing, the data do not need to be packed, therefore, at step S16, the data are normally output to the external bus.

In order to carry out the second conventional method, sets of addresses for effecting the packing are previously determined in address space and, when the former one of each set of the above addresses is accessed from the CPU, the data is held, and when the

latter one of the set is accessed, the two word size data are packed into one to effect a double word transfer. Here, however, the above addresses are previously determined to be accessed from only one process. In this method, the addresses for effecting the packing are previously determined in address space, therefore, there arises a problem that address space cannot be freely allocated to each process.

Alternately, the conventional methods have problems in that the allocation of address space corresponding to the external memory to each process (task) and access operations to the external memory are not freely determined.

Fig. 9 is a diagram showing a constitution of an interface of the embodiment of the present invention. The data processing apparatus of the present invention has a total constitution shown in Fig. 1, and the external memory is connected to the apparatus as shown in Figs. 3 and 6.

In Fig. 9, reference 12A indicates a data conversion unit; 12B indicates an address conversion unit; 121 indicates a first write buffer which holds upper 32-bit data of the 64-bit data output to the external data bus 311; 122 indicates a second write buffer which holds lower 32-bit data of the 64-bit data output to the external data bus 312; 123 inicates a first read buffer which latches upper 32-bit data of the 64-bit data on the external data bus 311; 124 inicates a second read buffer which latches lower 32-bit data of the 64-bit data on the external data bus 312; 125 indicates an address buffer which outputs address signals to the external address bus 321 when the external memory is accessed; 126-1 to 126-3 indicate packing buffers each of which holds 32-bit data on the internal data bus; 127 indicates a write multiplexer which selects data from among the data held in the packing buffers 126-1 to 126-3 and the data on the

internal bus 151 and outputs the selected data to the write buffer 121; 131-1 to 131-3 indicate read packing buffers each of which holds 32-bit data output from the second read buffer 124; 128 indicates a read multiplexer which selects data from among the data output from the first and second read buffers 123 and 124 and the data held in the read packing buffers 131-1 to 131-3; 129 indicates a first control circuit of the interface; and 130 indicates a discrimination circuit which discriminates whether the addresses for effecting the packing are accessed or not.

Reference numeral 134 indicates a control circuit of the address conversion circuit 12B; 132 indicates an address mapping table constituted by a Static RAM which has a capacity of, for example, 256 × 16 bits; and 133 indicates an address multiplexer for selecting address signals supplied to the address mapping table 132 between a group of upper eight bits a16 to a23 or a group of bits a2 to a9 of the internal bus address signals. When the group of the address bits a16 to a23 are selected, address space divisions A1 to A255 are selected. When the group of the address bits a2 to a9 are selected, AMT space is selected. When address space divisions A1 to A255 are selected, the address mapping table 132 is looked up with indices a16 to a23. When the AMT space is selected, contents of the address mapping table 132 can be read or rewritten by accessing addresses in the AMT space. The addresses to be accessed are determined by setting the states of the address bits a2 to a9. The address buffer 125 outputs, as an address of the external bus, the lower 16 bits of the address of the internal bus and the upper 16 bits output from the address mapping table 132 which is the result of address conversion of the upper 8 bits of the internal bus.

Fig. 10 is a diagram showing a memory map according to the embodiment of the present invention, wherein

reference numeral 31 indicates internal bus address space and 32 indicates external bus address space.

As shown in Fig. 10, in the internal address space, 64K of address space, from 000000 to 00ffff (hexadecimal), are allocated to a space in the address mapping table (AMT), 64K of address space, from 010000 to 01ffff, are allocated to the address space A1, 64K of address space, from 020000 to 02ffff, are allocated to the address space A2, 64K of address space, from 030000 to 03ffff, are allocated to the address space A3, and similarly hereinafter, 64K of address space from ff0000 to ffffff, are allocated to the address space A255.

In this embodiment, the address of the internal bus has 24 bits, the address of the external bus has 32 bits, and the total address spaces of the external bus are greater, by 256 times, than the address space of the internal bus. The size of the divided address spaces is the same between the internal bus and the external bus.

The address space of the external bus is divided into spaces B0 to B65535, and any value can be set as the upper 16 bits of the external bus in the address mapping table 132. Therefore, the address spaces A1 to A255 of the internal bus are mapped to any one of the address spaces B0 to B65535 of the external bus by the address mapping table. That is, by making an access from the CPU to the address space An (n is 1 to 255), in practice, access can be made to the address space Bm (m is 0 to 65535).

By providing the above-mentioned address mapping table 132, the problem of the above-mentioned second conventional method that the address space cannot be freely allocated to each process is resolved.

Fig. 11 is a diagram for explaining the address conversion in the address mapping table.

As described above, the address mapping table 132 is constituted by a static RAM of 256 × 16 bits, and

realizes a table of 256 entries × 16 bits. Entry numbers 1 to 255 of the address mapping table correspond to address spaces A1 to A255, and the entry number 0 is not used.

Then, as shown in Fig. 11, the upper 8 bits of the internal bus address are converted by the address mapping table 132 into the upper 16 bits of the external bus address, and the lower 16 bits of the internal bus address are directly output to the external bus address.

Fig. 12 is a diagram showing the constitution of the discrimination circuit 130. In Fig. 12, reference numeral 51 indicates an OR circuit; 521, 522, 523 and 524 indicate NOT circuits; and 531, 532, 533 and 534 indicate AND circuits. The discrimination circuit of Fig. 12 decodes the upper 8 bits a23 to a16 of the internal bus address, and discriminates between address spaces A8 to A255 without a packing function, AMT space, and address spaces A1 to A7 with a packing function.

That is, the discrimination circuit of Fig. 12 discriminates between the address spaces A8 to 255 without a packing function when [1] is included in any one of a19 to a23, discriminates the AMT space when a19 to a23 are all [0] and a16, a17 and a18 are [0], respectively, discriminates the address space A1 when a 16, a17 and a18 are [1, 0, 0], discriminates the address space A2 or A3 when a16, a17 and a18 are [0, 1, 0] or [1, 0, 1], and discriminates the address space A4, A5, A6 or A7 when a16, a17 and a18 are [0, 0, 1], [1, 0, 1], [0, 1, 1], [1, 1, 1].

Fig. 13 is a flowchart showing operations in the data processing apparatus according to the present invention. The operation of the embodiment will now be described with reference to Fig. 13.

The data conversion unit 12A waits for an access from the CPU at step S21. When an access is made, at step S22, the data conversion unit 12A discriminates whether the access is directed to the address spaces A1

to A7 with a packing function.

When the access is not directed to spaces A1 to A7 with a packing function, the control proceeds to step S26 where the address conversion is effected by the address conversion unit 12B, so that the internal bus address is converted into an external address. That is, eight bits a16 to a23 corresponding to the entry numbers of the address mapping table are fed to the address mapping table 132, the data read from the address mapping table 132 are used as the upper 16 bits of the external bus, and the lower 16 bits a0 to a 15 of the internal bus address are directly output as the lower 16 bits of the external bus. Then, at step S27, the data is transferred to the external bus and the control returns back to the step S21.

When the access is directed to the address spaces A1 to A7 with a packing function at the step S22, at step S23, it is judged whether or not the addresses contain the upper 32 bits of the 64 bits, namely, the addresses are (8n). When the addresses are (8n), at step S28, the data are held in the packing buffers 126-1 to 126-3 corresponding to the address spaces, and the control returns back to the step s21.

When the comparison of addresses at the step S23 indicates that the addresses are of the latter 32 bits of the 64 bits, namely, the addresses are (8n+4), the control proceeds to a step S25 where the data held in the packing buffers 126-1 to 126-3 are packed. The control then proceeds to the step S26 to execute the above-mentioned processing.

Fig. 14 is a timechart showing an example of operations of this embodiment. In this example, the transfer operations for 64-bit data are repeated two times. In each transfer operation, two 32-bit data are packed into 64-bit data, then the 64-bit data are transferred.

In Fig. 14, "CLK" indicates a clock signal of the

internal bus. Each signal of the internal bus is sampled at a positive edge of the clock signal. "*STRB" indicates a strobe signal of data signal and address signal of the internal bus. These signals are active when "*STRB" is "0", and are inactive when "*STRB" is "1". "A" indicates an address signal in the internal bus. "D" indicates a data signal in the internal bus. "RD" indicates data transfer direction. When "RD" is "1", the CPU reads data from the external memory. When "RD" is "0", the CPU writes data into the external memory. "*ACK" indicates a response to the data transfer in the internal bus. "WBUFFER" indicates data held in the packing buffer. "WDH" indicates data held in the first write buffer. "WDL" indicates data held in the second write buffer. "A BUFFER" indicates address signal output from the address buffer which is obtained by converting the internal address according to the address mapping table (AMT). "*WRITE" indicates indicate data transfer direction. When "*WRITE" is "0", data are written into the external memory. When "*WRITE" is "1", data are read from the external memory. "SIZE" indicates a control signal showing width of data. "*STROBE" indicates strobe signals of data signal and address signal of the external bus. These signals are active when "*STROBE" is "0", and are inactive when "*STROBE" is "1". "*DACK" indicates a response to the data transfer on the external bus.

As shown in Fig. 14, 32-bit data are output to the internal bus every three clock cycles. The first data DH(8N) is latched by the packing buffer at one clock cycle after the first data DH(8N) is output to the internal bus. The packing buffer holds the first data DH(8N) during six clock cycles. At one cycle after the second data DL(8N+4) is output to the internal bus, the first write buffer latches the first data DH(8N), and the second write buffer latches the second data DL(8N+4). The first and second write buffers hold and

output the latched data during six clock cycles. Further, at one cycle after the first and second data are output to the external data bus, the address buffer outputs the external bus address signals which are converted by using the address mapping table. The first and second data are written into an address indicated by the external bus address signals.

In the above embodiment, groups of write and read buffers corresponding to the address spaces A1, A2 and A3, A4 to A7 are respectively allocated to processes. However, for example, buffers are respectively provided for address spaces A1 to A7, and these buffers may be respectively allocated to different processes.

Moreover, the address spaces for effecting the packing may not correspond to the processes in a one-to-one manner. That is, when, for example, the first processing is using an address space that effects packing, the address space that effects packing but that is not used may be allocated to the second process thereby to dynamically allocate the address space that effects packing to the processing.

According to the present invention as described above, the address space is divided and is allocated to processes and, besides, an address conversion function is provided to effect the mapping from the address space to another address space. This makes it possible to realize packing processing in a multitasking environment. Moreover, the address conversion mechanism makes it possible to avoid the problem of a limitation on the accessible addresses imposed by the division of the address space.

C L A I M S

1. A multitasking data processing apparatus for concurrently executing a plurality of processes in a time-sharing method, comprising:

an internal bus; and

an interface to which an external memory is connected in a form that the external memory is directly accessible from a central processing unit of the apparatus via an external bus having data bus width different from that of the internal bus;

said interface comprising:

data signal conversion means for converting data signals when said external memory is accessed; and

address signal conversion means for converting address signals when said external memory is accessed;

said address signal conversion means comprising:

an address mapping table indicating correlations between the internal bus addresses and the external bus addresses, said correlations being optionally settable.

2. A multitasking data processing apparatus as set forth in claim 1, wherein said external memory is accessible by a unit of the data bus width of said internal bus.

3. A multitasking data processing apparatus as set forth in claim 1 or 2, wherein said data bus width of said external bus is an integer multiple of the data bus width of said internal bus.

4. A multitasking data processing apparatus as set forth in claim 3, wherein said data conversion means includes data packing means for packing internal bus data, and a plurality of internal bus data are simultaneously written into said external memory after packing.

5. A multitasking data processing apparatus as set forth in claim 4, wherein said data conversion means includes a plurality of data packing means.

6. A multitasking data processing apparatus as set forth in claim 5, wherein the address space of said external

memory is divided into a plurality of blocks, and said plurality of data packing means are respectively allocated to said blocks.

7. A multitasking data processing apparatus as set forth in claim 3, wherein said data conversion means includes a plurality of data buffers for time-dividedly outputting external bus data into said internal bus when data is read from said external memory.

8. A multitasking data processing apparatus as set forth in claim 7, wherein said data conversion means includes a plurality of secondary data buffers for instantaneously holding data of one of said data buffers.

9. A multitasking data processing apparatus as set forth in claim 8, wherein address space of said external memory is divided into a plurality of blocks, and said plurality of secondary data buffers are respectively allocated to said blocks.

10. A multitasking data processing apparatus as set forth in claim 6 or 9, wherein said blocks of said external memory are dynamically allocated to said processes.

11. A multitasking data processing apparatus as set forth in any one of the preceding claims, wherein said external memory is shared memory.

12. A multitasking data processing apparatus substantially as described with reference to Figures 1, 2, and 9 to 14 of the accompanying drawings.

Patents Act 1977
Examiner's report to the Comptroller under Section 17
(The Search report)

-27-

Application number
GB 9413165.3

Relevant Technical Fields

(i) UK Cl (Ed.M) G4A AFD, AFGDC, AKA, AKB1, ANV

(ii) Int Cl (Ed.5) G06F 12/10, 13/40

Search Examiner
PAUL NICHOLLS

Date of completion of Search
25 AUGUST 1994

Databases (see below)
(i) UK Patent Office collections of GB, EP, WO and US patent specifications.

(ii) ONLINE DATABASE: WPI

Documents considered relevant following a search in respect of Claims :-
1-12

Categories of documents

X: Document indicating lack of novelty or of inventive step.

Y: Document indicating lack of inventive step if combined with one or more other documents of the same category.

A: Document indicating technological background and/or state of the art.

P: Document published on or after the declared priority date but before the filing date of the present application.

E: Patent document published on or after, but with priority date earlier than, the filing date of the present application.

&: Member of the same patent family; corresponding document.

Category	Identity of document and relevant passages	Relevant to claim(s)
A	EP 0194696 A2 (SONY) whole document	

Databases: The UK Patent Office database comprises classified collections of GB, EP, WO and US patent specifications as outlined periodically in the Official Journal (Patents). The on-line databases considered for search are also listed periodically in the Official Journal (Patents).